1-1-2002

Classification of ultrasonic signals using hidden Markov models

Melinda Sue Vander Velden
Iowa State University

Classification of ultrasonic signals using hidden Markov models

by

Melinda Sue Vander Velden

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Electrical Engineering

Program of Study Committee:
Julie Dickerson (Major Professor)
John Bowler
Vasant Honavar

Iowa State University

Ames, Iowa

2002

Graduate College
Iowa State University

This is to certify that the master's thesis of

Melinda Sue Vander Velden

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

Table of Contents

List of Figures

List of Tables

Abstract

Submarine hulls consist of large steel sections welded together. During the welding process and with use, defects can develop in the welds. The seriousness of a flaw can be determined by cutting into the weld or by stressing the material to its breaking point. These testing methods, while accurate, destroy the material. Ideally, the flaw should be located and classified without destroying the material.

Weld defect classification methods used to date generally involve reducing the dimensionality of the data through the selection of a set of features from the data such as the Fourier coefficients, wavelet coefficients, moments, or principal components. These features have then been classified using neural networks. This approach has produced unsatisfactory results. Introduced by this research is the application of hidden Markov models (HMMs) and support vector machines (SVMs) to classify weld defect data.

HMMs provide a probabilistic method for generating defect models and predicting the class of test defect sequences. HMMs have been used successfully in many pattern recognition problems such as speech and handwriting recognition. Experimentally, HMMs achieved 60% correct classification. HMMs makes the Markovian assumption of strictly previous state dependence. The optimal number of states must be determined experimentally, and the number of states necessary may be dependent on the class being learned.

SVMs belong to a class of machine learning algorithms known as kernel machines. SVMs exploit information about the inner products between data, and are capable of generating complex decision boundaries. SVMs attained 75% correct classification, indicating that there is potential for the data to be classified accurately. With additional research, HMMs may prove capable of more accurate weld defect classification, but initial results indicate that SVMs are better suited to the problem.

Chapter 1. Introduction

1.1. Problem Definition

Submarine hulls consist of large steel sections welded together. During the welding process and with use, defects can develop in the welds. The seriousness of the flaw can be determined by cutting into the weld or stressing the material to its breaking point. These testing methods, while accurate, destroy the material. Ideally, the flaw could be classified without destroying the material. Some defects are serious, adversely affecting structural integrity, and must be repaired immediately. However, since not all flaws require repair, determining the location and severity of flaws non-destructively would be beneficial.

One effective method used for this purpose is X-ray, but this method poses safety issues. A safe alternative to X-ray is ultrasound. Ultrasonic non-destructive evaluation (NDE) is also more convenient because the testing equipment is portable and the results are available immediately. This thesis investigates the potential of ultrasonic NDE to classify defects in welds. The data for this project consists of several signals of ultrasonic scans of weld defects which have been classified using X-ray or destructive investigation. Unique to this thesis is the application of hidden Markov models to classify these signals. Principal component analysis and support vector machines are also used to evaluate the classification potential of this data.

1.2. Organization of Thesis

Chapter 2 contains background information on the problem of classifying weld defects using ultrasonic NDE. Brief summaries of related research that others have conducted are also provided. Chapter 3 provides descriptions of the methods used in this project. Chapter 4 presents the results of hidden Markov model classification. Chapter 5 is a discussion of the most significant results with comparisons to classification using principal component analysis and support vector machines. Additionally, recommendations for future work are provided.

Chapter 2. Background

The Materials Assessment and Research Group (MARG) at Iowa State University researches techniques for detecting flaws in materials and determining the categories and severity of these defects without damaging the material under test. Perhaps the best known project undertaken by MARG is the study of defects in natural gas transmission pipelines. Millions of miles of gas pipelines are buried below the ground in the United States. Defects in these pipelines have led to explosions and loss of life, emphasizing the need for quickly and accurately identifying and repairing such problems. Defect detection is accomplished by running a pipeline inspection gadget (PIG) through the pipelines. The PIG is propelled by the natural gas itself, and brushes on the PIG create a magnetic field within the pipe. Information is stored within the PIG, and defects are identified when this information is analyzed. A number of techniques can be used to identify irregularities in the magnetic field which often indicate defects. Once the location of a defect is known, only that section of the pipeline needs to be unearthed and replaced or repaired.

Another MARG project involves testing the quality of welds in submarine hulls and reactor welds. The reactors in submarines are surrounded by massive pieces of steel fused together by welds. Whenever repairs or maintenance are required in the reactor, the welds in this metal shield must be cut. When the shield is fused again, four types of problems can occur. The weld may fail to adhere to the metal, which is called a lack of sidewall fusion or simply a lack of fusion defect. A crack can also form within the weld. Both lacks of fusion and

cracks are known as planar flaws. Another type of defect that can be found in welds is a porosity, which occurs when pockets of air or another gas are included in the weld. Slags are similar to porosities, and occur when dirt, metal, or other solid impurities are incorporated in the weld. Both porosities and slags are known as volumetric defects.

The problems presented by both of the above mentioned projects can be solved using non-destructive evaluation (NDE) techniques.

2.1. Non-Destructive Evaluation

Non-destructive evaluation (NDE) is a process of testing a material without rendering it useless. An everyday example that illustrates the basic need for NDE is the process of selecting fresh fruit at a grocery store. One way of determining the quality of a piece of fruit is to simply take a bite out of it. This approach unequivocally informs the person if the fruit is good. Unfortunately this test ruins the fruit.

It would be much better to have a non-destructive method for determining if produce is good. One can gently squeeze the fruit to determine if it is rotten. The color indicates the ripeness. If there is a smell, mold, or insects present, the fruit is past its prime. All of these methods indicate the quality of the fruit, but do not destroy it in the process.

Testing an item without damage is the motivation behind NDE. In every application of NDE, the tester applies some form of energy to the object, measures the response to this

applied energy, and makes a decision based on the response. A diagram outlining an NDE system is shown in Figure 2.1.

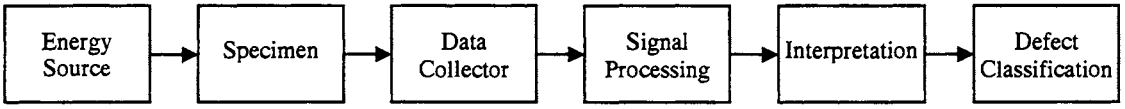

Figure 2-1. Non-destructive evaluation system

The energy source induces a signal into the specimen through a transducer. The properties of the specimen alter the signal, and this response signal is collected by a transducer. The collected signal may require processing such as denoising, normalization, or gating to include only the relevant information. Interpretation of a processed signal involves selecting features that differentiate one class from another. Interpretation is followed by classification, in which the features are used to determine to which class each signal belongs.

There are many energy sources that can be used in NDE, including radiography (X-ray) [1, 2], ultrasound [3, 4], eddy current [5], and visual inspection [6, 7]. For this thesis, an ultrasonic energy source was used.

2.2. Ultrasonic Non-Destructive Evaluation

Ultrasonic NDE uses a very high frequency sound wave as the energy source. Ultrasound is useful for NDE because it can travel through many solid materials more readily than X-rays

and other forms of electromagnetic energy. Ultrasonic testing is also capable of locating small defects embedded several feet in a piece of metal.

Ultrasonic waves obey many of the laws of physics that apply to light. As the ultrasonic wave passes between similar materials, it is refracted. If the properties of the material the sound wave is passing through change significantly, the wave is reflected. This characteristic is valuable to NDE because the wave will travel through the test specimen, but will be reflected off the far wall of the specimen rather than continue to travel through other materials such as the surface the specimen is resting on. Figure 2.2 shows a typical ultrasonic NDE situation.

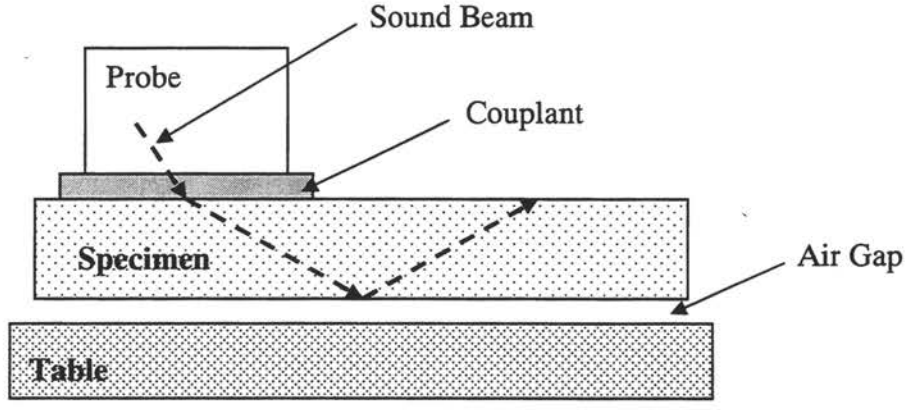

Figure 2-2. Ultrasonic NDE system

At low frequencies, a sound wave travels in all directions rather than in a well defined beam. As the frequency increases, the wave becomes less diffused and more beam-like. The fact that ultrasound can travel in well defined beams is another reason ultrasonic waves are useful

for NDE. For materials testing, the frequency range of 0.5 to 20 MHz is used. For most homogeneous engineering materials, the frequency is within the range of 1 to 10 MHz [3, 4]. For this project, a frequency of 2 MHz was used in all cases.

The sound wave originates from the probe and refracts as it passes into the specimen under test. The particles in the specimen vibrate, transferring the energy from one particle to another. Some materials are better at transmitting ultrasound than others. Ultrasound does not travel through air well, so a couplant is needed to transfer the maximum amount of energy into the specimen. Oil, water, and commercial couplants are examples of mediums that help the beam jump the air gap between the probe and the specimen. Because ultrasound will not travel through the air gap, the sound wave reflects off the far surface of the specimen rather than continue into the table. Likewise, if there is a defect in the specimen, the sound wave will reflect off the defect rather than continue through the specimen.

A transducer is needed to transmit the ultrasonic waves, as is a method for measuring the reflections. There are two types of transducers: magnetostrictive and piezo-electric. A magnetostrictive transducer takes advantage of the dimensional change produced in ferromagnetic materials when they are subjected to an alternating magnetic field. A piezo-electric transducer subjects crystals of minerals, such as natural quartz, to an alternating electrical potential. This causes the crystals to expand and contract causing vibration. Likewise, when the crystals are subject to a mechanical vibration, an electrical potential is produced. This is known as the piezo-electric effect. A piezo-electric transducer may contain a single element that both transmits ultrasonic pulses and receives the reflected

energy, or it may contain independent transmission and reception elements. Single element transducers are more commonly used in flaw detection. Twin element transducers are good for detecting defects that are very close to the surface of the material under test and for detecting defects in materials whose surfaces are not smooth.

Proper angling of the transducer is essential to ensure that the beam hits the largest cross section of the defect perpendicularly. If a defect is perpendicular to the sound beam, the reflection will come back to the same place the beam was injected. However, if the defect is not perpendicular to the sound beam, the reflection may not be seen. The probe needs to be moved around on the specimen until the sensor lies in the path of the reflection from the defect. For this project, the transducer was fixed at a 60° angle to the surface of the material. [3, 4]

The intensity of the reflection, measured in decibels (dB), conveys information regarding the size of the defect. Another reason ultrasound is so useful for materials testing is that ultrasonic waves can travel long distances through many materials. Attenuation can occur as the reflection travels back through the material. The intensity of the reflection decreases the farther it travels through the material. This problem can make it difficult to detect small defects.

If the test material is of uniform composition and structure, the speed of the ultrasonic signal through the material is constant. By knowing the speed of the sound beam and the time of travel, one can calculate the distance the beam has traveled. The distance traveled represents

the thickness of the test material, and this time of flight test is often used to detect inconsistencies in thickness that may indicate corrosion in pipes and chemical vessels.

Ultrasonic testing can detect most internal defects in many solid materials such as steel, aluminum, copper, brass, and plastic. Ultrasound can also travel readily through liquids such as water and oil.

Another advantage of ultrasonic NDE is the immediate availability of the test results. Additionally, the testing equipment is portable, so inspection can be done on site, and there is no danger to the technicians, unlike X-ray techniques. [3, 4]

2.3. Weld Defect Classification

Welding is a method of fusing two pieces of metal together to act as one unit. The metal surfaces as well as a filler material are heated to a high temperature, at which the filler and contact areas of the metal surfaces become liquid. When heat is removed, the liquid metal returns to the solid state and the two pieces are joined. A weld is depicted in Figure 2.3. Welding is a common practice to fuse together parts of natural gas transmission pipelines, automobiles, hulls of ships, and other metal objects.

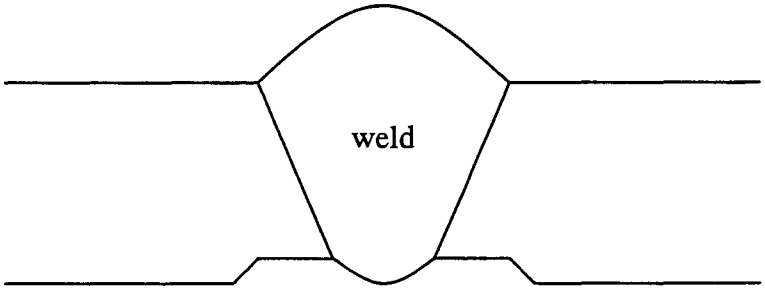

Figure 2-3. Typical weld

Four classes of defects can occur in welds: cracks, lacks of sidewall fusion, porosities, and slags. Cracks occur when a split develops in the weld. A lack of fusion occurs when the weld does not adhere the metal surfaces. Porosities are small pockets of air or other gas that become entrapped within the weld. Slags are pockets of solid impurities that become entrapped in the weld. Cracks and lacks of fusion are known as planar flaws, and are the most serious defects as they can allow water to penetrate the submarine hull or radiation from the reactor to escape. Porosities and slags are known as volumetric flaws, and while they are not as serious as planar flaws, they weaken the weld and could later develop into cracks. It is vital to repair planar flaws, but since it is very costly to repair welds in submarine hulls, volumetric flaws do not require immediate attention.

X-ray techniques can successfully distinguish between the defect types. While X-ray NDE is an effective way to find critical defects, this method presents safety issues. Other workers must evacuate the area to avoid the radiation, and there is potential danger for the person performing the test. Ultrasonic NDE is a safer alternative. Evacuation is not necessary when ultrasonic testing is taking place, and there is no threat to the technician.

The ultrasonic signals are collected by moving a transducer across the surface of the weld and surrounding metal. This data can be represented in three forms: A-scans, B-scans, and C-scans. [3, 4]

An A-scan is a signal strength versus time representation for one transducer location, and can be used to estimate the depth and size of a defect. Some examples of crack A-scans are shown in Figure 2.4, lack of fusion A-scans are shown in Figure 2.5, porosity A-scans are shown in Figure 2.6, and slag A-scans are shown in Figure 2.7. While differences and similarities may be observable in these particular signals, overall there are no obvious features that distinguish the four defect classes of this data set.

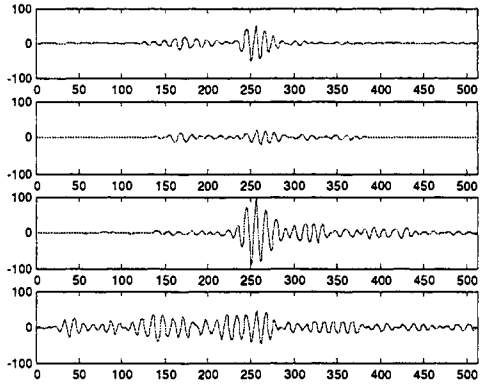

Figure 2-4. Examples of crack A-scans

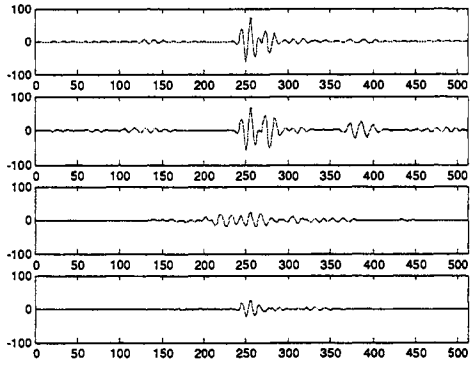

Figure 2-5. Examples of lack of fusion A-scans

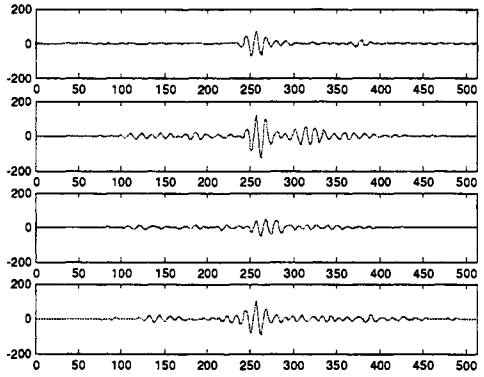

Figure 2-6. Examples of porosity A-scans

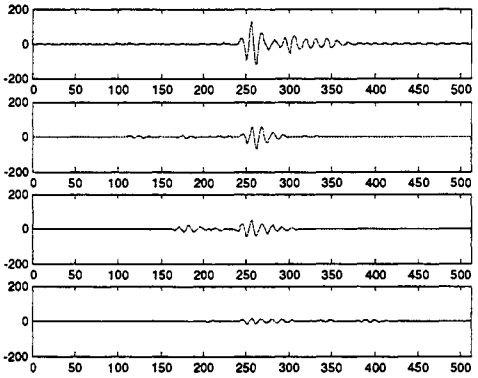

Figure 2-7. Examples of slag A-scans

A B-scan, which is a two-dimensional image representation, provides a cross-sectional view of the weld, and consists of several A-scans. B-scans provide front and back views of the surface of the material under test, and can be used to determine the relative location of a defect within the material. Similar to a B-scan is a B'-scan. Both scans provide cross sectional views normal to the surface of the weld, but a B-scan is an axial scan and a B'-scan is a circumferential scan.

A C-scan is a two-dimensional image that depicts a three-dimensional weld. Each point of the image summarizes an A-scan by using the peak amplitude of the A-scan taken at that point on the weld, and in this way it is similar to a radiograph. A C-scan display outlines the contour of the defect, but does not reveal the depth of the defect within the test material. Typical B-, B'-, and C-scans are shown in Figure 2.8. [8]

For this thesis, classification is based solely on A-scans.

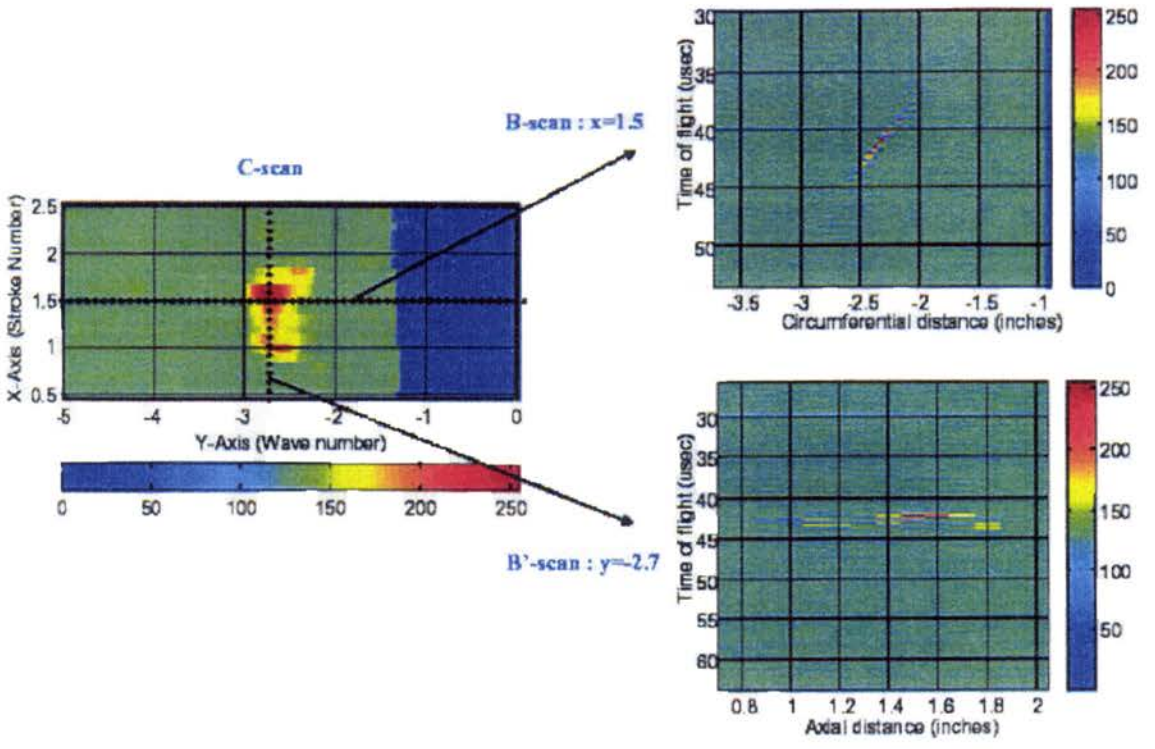

Figure 2-8. Typical B-, B'-, and C-Scans

2.4. Weld Defect Classification Methods

The major steps of automatic signal classification are preprocessing, feature extraction, and classification. In the preprocessing phase, denoising, windowing, and gating are typical procedures. The goal of feature selection is to reduce the dimensionality of the data by selecting characteristics that describe or approximate the original signal with a smaller number of terms. In the classification phase, each signal is assigned a class designation.

Several preprocessing procedures, features, and classification schemes have been researched for use in this project including principal component analysis (PCA), moment analysis, and frequency domain transformations such as the fast Fourier transform. The major research efforts are summarized in this section.

2.4.1. Synthetic Aperture Focusing Technique

Synthetic aperture focusing technique (SAFT) is an algorithm that can be used to reduce the signal to noise ratio of defect signals, removing noise and leaving the amplitude unchanged. Surface signals are minimized, so there is less signal loss. SAFT has been applied to ultrasonic scans of welds and other materials as a preprocessing step with the goal of improving classification. [9, 10]

2.4.2. Fourier Analysis

Fourier analysis provides representations of the frequency content of a signal. The frequency components obtained from Fourier analysis can be used as features for pattern recognition. The Fourier transform maps a time domain signal to the frequency domain using complex exponential basis functions.

The discrete Fourier transform can be applied to a discrete function that consists of N samples. The discrete Fourier transform is defined by the following equation:

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\exp\left(\frac{-j2\pi ux}{N}\right)$$

where $u = 0, 1, 2, \ldots, N-1$.

The discrete Fourier transform can also be extended to two dimensions, and the transform is defined by the following equation:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\left(-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right)$$

where $u = 0, 1, 2, \ldots M-1$ and $v = 0, 1, 2, \ldots, N-1$.

The fast Fourier transform (FFT) is an algorithm that reduces the number of multiplications and additions to $O(N \log N)$. [11, 12, 13, 14]

Fourier analysis has been applied to the weld defect classification problem. The Fourier coefficients were calculated in the feature selection stage and classified using a neural network. [10]

2.4.3. Wavelet Transform

The wavelet transform can also be used in the feature selection phase. The wavelet transform is defined in terms of basis functions that are obtained by compressing or dilating and shifting a mother wavelet. Wavelets are another way to achieve localization, (*i.e.* signals in both x and u), as discussed in the Gabor transform section. [15, 16, 17].

Wavelet analysis has been applied to the weld defect classification problem. The wavelet coefficients were calculated in the feature selection stage and classified using a neural network. [10]

2.4.4. Discrete Cosine Transform

The one-dimensional discrete cosine transform (DCT) is defined by the following equation:

$$C(u) = \alpha(u)\sum_{x=0}^{N-1} f(x)\cos\left(\frac{(2x+1)u\pi}{2N}\right)$$

for $u = 0, 1, 2, \ldots, N-1$. The variable α is $\sqrt{\frac{1}{N}}$ for $u = 0$ and $\sqrt{\frac{2}{N}}$ for $u = 1, 2, \ldots, N-1$.

The two-dimensional DCT is defined by the following equation:

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{2N}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right)$$

for $u, v = 0, 1, 2, \ldots, N-1$, and α is defined the same as in the one-dimensional case. [11]

The DCT has been applied to the weld defect classification problem. The DCT coefficients were calculated in the feature selection stage and classified using a neural network. [10]

2.4.5. Moment Analysis

A moment is a statistical property of a function. Mean and variance are examples of moments. A relatively small number of moments of one or two dimensional signals can be used as features.

If the function is a discrete valued image $I(m,n)$, as is the case with a B-scan or B'-scan image, the moments are defined by the following equation:

$$M_{pq} = \sum_{m} \sum_{n} I(m,n) m^p n^q$$

As in the continuous case $p, q = 0, 1, 2, \ldots$ in the above equation. Given a continuous function with finite discontinuities, $f(x,y)$, or a discrete image, $I(m,n)$, M_{pq} characterizes the function uniquely. This fact allows one to use a finite number of moments as features for classification.

The centralized moments can also be used as features for classification. The coordinates of the centroid of the image $I(m,n)$ are $(\overline{m}, \overline{n})$, where $\overline{m}$ is the average of the x-coordinates and $\overline{n}$ is the average of the y-coordinates. The centroid is defined by the following two equations:

$$\overline{m} = \frac{M_{10}}{M_{00}}$$

$$\overline{n} = \frac{M_{01}}{M_{00}}$$

The centralized moments can then be defined by the following equation:

$$C_{pq} = \sum_{m}\sum_{n}(m-\overline{m})^{p}(n-\overline{n})^{q}$$

As before, $p, q = 0, 1, 2, \ldots$ in the above equation. Often the notation m_{pq} is used to represent the centralized moment. C_{pq} was used in this case to represent the centralized moments to avoid confusion with the m coordinate of a two dimensional image.

The moment concept can be further expanded to define another feature, normalized central moments. To define normalized central moments, first define the quantity r as follows:

$$r = \frac{p+q}{2} + 1$$

In the above equation, $p + q = 2, 3, 4, etc.$ The normalized central moment is defined by the following equation:

$$\mu_{pq} = \frac{C_{pq}}{C_{00}^{r}}$$

Typically when using normalized central moments as features, $p + q \leq 7$. Higher order moments are generally uninformative.

Moment analysis has been applied to the weld defect classification problem. Several moments were calculated for several B-scan and B'-scan images in the feature selection stage and classified using a neural network. [10]

2.4.6. Principal Component Analysis

Principal component analysis (PCA) is a classical technique whose goal is to reduce the dimensionality of a data set by transforming to a new set of variables, the principal components, which summarize the important features of the data. PCA is useful when one has gathered a large number of signals that contain redundancy due to the fact that signals of the same type are correlated. Ideally the variation in the data can be explained by a few principal components rather than all of the original variables. PCA can be used in conjunction with other multivariate techniques such as multiple regression analysis and discriminant analysis.

The principal components are linear combinations of the original data. The original data set is given by $\{x_1, x_2, \ldots, x_p\}$. The k^{th} principal component is given by the following equation:

$$z_k = \sum_{i=1}^{p} a_{ki} x_i$$

The a_{ki}'s are weights that are chosen to maximize the variation of each principal component. There are a maximum of p principal components that can be used in further analysis. Using all p principal components would preserve all of the information of the original data. However, this would defeat the purpose of PCA to reduce the number of variables.

To perform PCA, first generate the covariance matrix of the original data. The correlation matrix can be used as an alternative. The weights, a_{ki} from the above equation, are the eigenvectors of the covariance matrix. The principal components are projections, and are calculated using the above equation, which is equivalent to the dot product.

The principal components are uncorrelated, and are ordered by eigenvalues such that the first principal component has the largest variance of all the principal components, the second principal component has the second largest variance, and so on. The last principal components have the smallest variances, and generally represent noise in the data. Ideally, the first few principal components should capture most of the information in the original data. This means that the first few principal components should sum to be approximately p. If this is not the case, clustering results will be poor, and several classification techniques cannot be used to separate the data into classes. Neural networks are an example of a tool that cannot be used when PCA suggests the data is not separable.

PCA has also been applied to ultrasonic scans of weld defects with the goal of clustering the data into either four defect classes (crack, lack of fusion, porosity, and slag) or two classes (planar and volumetric). It should be noted that the research in [8] was not performed on the same data set as was used in this project, but the two data sets are similar. [8, 18, 19, 20]

2.4.7. Multilayer Perceptrons

Neural networks are capable of generalizing complex decision boundaries, and their parallel structure allows neural networks to make decisions very quickly.

Neural nets are trained with a set of patterns of known types in a trial, error measurement, and adjustment process. The neural net is then tested on another sample, and errors are

detected and corrected. This iterative process of designing, testing, and redesigning continues until the training goals are met.

The most commonly used type of neural network is the multilayer perceptron (MLP), and an MLP with one hidden layer is shown in Figure 2.9.

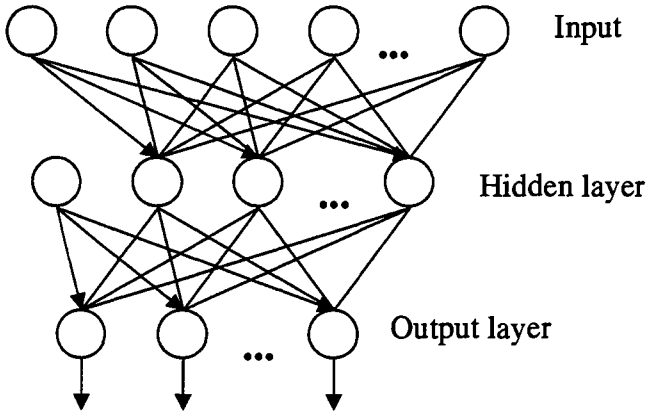

Figure 2-9. Multilayer perceptron

An MLP has multiple inputs, at least one hidden layer, and multiple outputs. Each layer is made up of several simple units called neurons. Each neuron sums its inputs and passes the sum through a non-linear excitation function. A commonly used excitation function is the sigmoid function which is given by the following:

$$y = \frac{1}{1+e^{-x}}$$

The connections between neurons have weights representing the interconnection strengths. The MLP is first put through a training phase in which it approximates the relationships

between the inputs and the corresponding outputs in training data set. These relationships are stored in the weights. [21, 22]

2.4.8. Probabilistic Neural Networks

A probabilistic neural network (PNN) is based on Bayesian decision theory. Similar to the MLP (Section 2.4.7), PNN's are capable of generalizing complex decision boundaries and making decisions quickly. Additionally, under certain easily met conditions, the decision boundaries generated by a PNN approach the Bayes optimal decision surface. Bayes strategies are classification schemes that classify patterns such that the expected risk is minimized. Like an MLP, a PNN requires supervised training and is a statistical classification method. A PNN is unique in the fact that it uses exponential activation functions rather than sigmoidal activation functions.

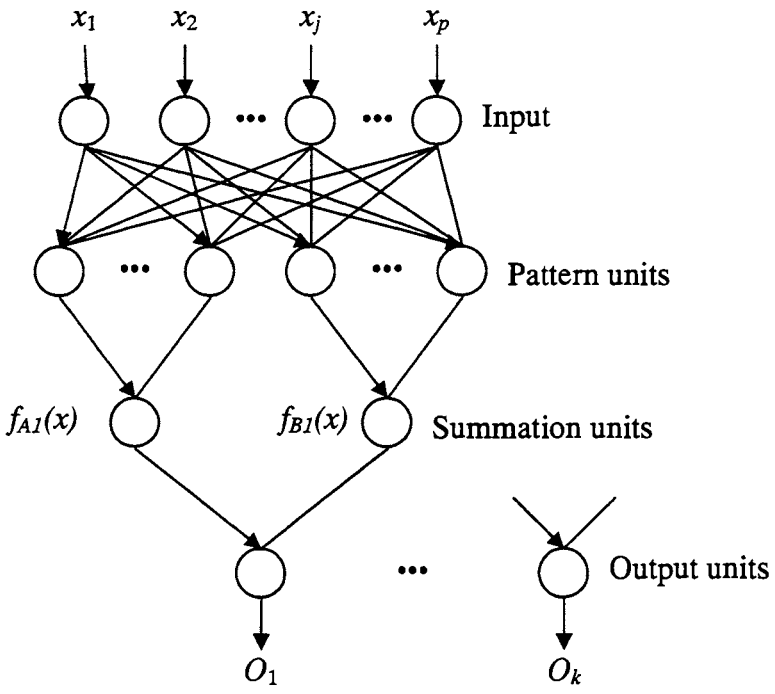

Figure 2-10. Probabilistic neural network

A PNN has the advantages that the output is probabilistic, and it can be trained faster than the back-propagation procedure used for MLPs. Additionally, erroneous samples can be tolerated, sparse samples can be used without sacrificing network performance, and PNNs operate in parallel without a need for feedback from individual neurons to the inputs. PNNs have the disadvantage of a large network size. [23]

2.4.9. Adaptive Resonance Theory Networks

Adaptive resonance theory (ART) networks are designed to overcome the stability–plasticity problem. Stability is the ability of a network to prevent interactions between its stored patterns. Plasticity is the ability of a network to learn new patterns in response to new

stimuli. The ART learning algorithm is similar to a clustering algorithm. There are four types of ART networks. ART-1 networks are for use with binary patterns. ART-2 networks are for analog patterns. Fuzzy ART and ARTMAP are supervised ART networks. For details about ART networks, refer to [22].

2.4.10. Support Vector Machines

Support vector machines belong to a large class of learning algorithms known as kernel machines. Kernel machines exploit information about the inner products between data, and are capable of classifying complex patterns. Linear classifiers are not capable of dealing with non-linearly separable data or noisy data. The goal of support vector machines is to map non-linearly separable data into some feature space where they can be separated by a linear classifier.

The inner product between vectors **x** and **y** is defined by the following:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_i \mathbf{x}_i \mathbf{y}_i$$

The decision boundary is a hyperplane defined by:

$$f(\mathbf{x}) = \sum_{i=1}^{M} y_i \alpha_i \cdot k(\mathbf{x}, \mathbf{x}_i) + b$$

where **x** is one of M defect samples, y is the set of class labels corresponding to **x**, $k(\cdot,\cdot)$ is a kernel function, b is a bias term, and the sign of $f(\mathbf{x})$ determines the class of **x**. Any vector $\mathbf{x}_i$ that corresponds to a nonzero α_i is a support vector.

In the simples case, the kernel function is a simple dot product in the input space. The multilayer perceptron algorithm discussed in section 2.4.7 provides such an example. A non-linear SVM projects the samples into a highly dimensional feature space using a nonlinear mapping function and then constructs a hyperplane in this feature space. [24, 25]

Chapter 3. Methods

Most of the methods used so far by researchers in the Materials Assessment Research Group (MARG) have involved selecting significant features from the set of weld defects. Commonly used features include the Fourier transform (section 2.4.2), wavelet transform (section 2.4.3), and discrete cosine transform (section 2.5.4). A well chosen feature set captures the significant distinctions of the signals with minimal loss of information. These features are then used as the input of a neural network which performs the classification.

The use of hidden Markov models (HMMs) differs from the previously used methods because they use probabilistic models to predict a test defect signal's class. The model parameters are often not known *a priori* in pattern recognition problems such as the weld defect problem, but these parameters can be estimated from the training data using an iterative process called the Baum-Welch algorithm (section 3.2.1.7).

A further distinction between HMMs and other feature selection methods is that HMM algorithms are capable of classifying the test signals rather than turning the feature set over to a neural network or other classification scheme. Once a set of model parameters has been estimated for each defect type, the models can be used to calculate scores for the test sequences. For a given test sequence, a score that quantifies the probability that a model generated the test sequence is calculated for each of the defect types. The model that is most

probable is the final classification of the test sequence. Classification is accomplished using the forward algorithm (section 3.2.1.2).

3.1. Markov Chains

A Markov model is a simply a set of states and transitions between the states. A diagram describing a simple Markov process with two states is shown in Figure 3.1.

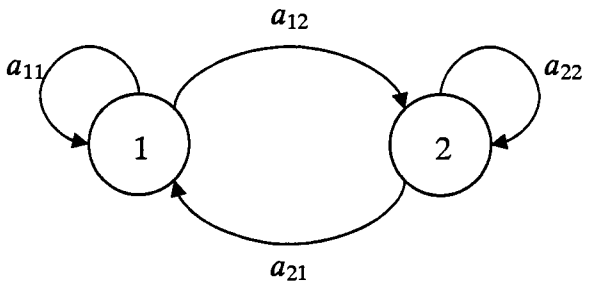

Figure 3-1. Two state Markov process

The a's in Figure 3.1 are the state transition probabilities where a_{11} is the probability of remaining in state one from one time instance to the next, a_{12} is the probability of making the transition from state one to state two, *etc.* Consider the example of a coin toss in the context of Figure 3.1. State one means "heads" was the result of the toss and state two means "tails" was observed. State transitions can be presented in matrix form as shown by the following example:

$$\mathbf{A} = \begin{bmatrix} 0.6 & 0.4 \\ 0.7 & 0.3 \end{bmatrix}$$

The matrix **A** gives the probability of moving from any state to any other state. The rows represent the previous state, and the columns represent the current state. From the above matrix it can be seen that the probability of observing "heads" given that the previous state was also "heads" is 0.6 for this particular coin. The probability of observing "heads" on this coin given that the previous state was "tails" is 0.7. Note that the rows must sum to one, but the columns might not sum to one.

Ordinarily, developing a probabilistic description of an observed process with several states would involve the use of the probabilities of all the preceding states. With a Markov process, only the current and one predecessor state are needed to develop a probabilistic description. For example, using the simple coin toss example from above, what is the probability of observing the observation sequence O = HHTTHTH (*i.e.* what is P(O))? To answer this question, an additional piece of information is needed: the initial state probabilities. The initial state probabilities are simply the probabilities of being in a particular state for the first observation. For the coin toss example, the initial state probability matrix is represented by the following:

$$\boldsymbol{\pi} = \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}$$

The probability of being in state one for the first observation is 0.6, and the probability of being in state two for the first observation is 0.4. Note that the sum of the initial state probabilities must be one. Now that both necessary model parameters, **A** and π, have been defined, P(O) can be calculated:

$$P(O) = \pi_1 * a_{11} * a_{12} * a_{22} * a_{21} * a_{12} * a_{21}$$

$$= (0.6)(0.6)(0.4)(0.3)(0.7)(0.4)(0.7)$$

$$= 8.4672 \times 10^{-3}$$

There are three parameters that characterize a Markov process. N is the number of states, which is the same as the number of observations. In the coin toss example discussed in this section, N is two. The other two parameters are the state transition probability matrix, $\mathbf{A}$, and the initial state probability matrix, π.

3.2. Hidden Markov Models

In the previous section, Markov models were considered. There was a one to one correspondence between the states and the observations. If "heads" were observed, one knew that the process was in state one, and if "tails" were observed the process was in state two. Only three parameters were required to characterize the model, N, $\mathbf{A}$, and π. For a hidden Markov model, there is not a one to one correspondence between the states and the observations, and the states are unobservable. All that is known is a set of observations, and the observer wishes to make assumptions about the underlying state sequence that produced the observation sequence.

The coin toss example discussed in section 3.2 can be extended to illustrate hidden Markov models. An observer and an experimenter are in a room, but are separated so the observer cannot see what the experimenter is doing. The observer knows that the experimenter is performing a coin toss experiment, but does not know how many coins are being tossed. The

observer is given only a series of observations: "heads" or "tails". If one assumes that only one coin is being tossed, the experiment is identical to that discussed in section 3.2.

Now assume there are two coins in the experiment, and only one coin is fair. The experimenter first chooses a coin by some unknown process, tosses it, and then gives the observer the observation from the coin. The observer does not know which coin the observation has come from, only if the result of the toss is "heads" or "tails". In this case, state one corresponds to coin one, state two corresponds to coin two, and the state sequence is unknown to the observer. Knowing the observation sequence does not tell the observer what the state sequence is, but the observer can infer some information about the state sequence based solely on his observations. This example can be extended to any number of coins. The number of states is the number of coins.

For a Markov chain, three parameters can completely characterize the model. Five parameters are required to define a hidden Markov model. First is N, the number of states. In the coin toss example, there are as many states as there are coins. The second parameter is M, the number of observation symbols that can be observed in each state. For the coin toss example, M is two regardless of the number of coins used in the experiment. There are two possible observations: "heads" and "tails".

The third parameter $\mathbf{A}$ is the state transition probability matrix which was discussed in section 3.2. $\mathbf{A}$ describes the probabilities of moving from any state to any other state. As

discussed before, the rows correspond to the previous state and the columns correspond to the current state. The probability a_{ij} is the probability of moving from state i to state j.

The fourth parameter is the emission probability, $e_j(k)$, and is the probability of observing symbol k while in state j.

The fifth parameter needed to define a hidden Markov model is π, the initial state probability matrix. For each possible state, this parameter defines the probability of being in that state for the first observation.

3.2.1. The Three Problems of Hidden Markov Models

Three basic problems exist for hidden Markov Models. In this section, each problem is defined, an example is given, and the solution is discussed. The first and third problems are of special interest in the context of this thesis as the third problem deals with model parameter estimation and the first problem deals with classification.

3.2.1.1. The First Problem for Hidden Markov Models

The first problem for hidden Markov models is to determine the probability of observing a given sequence when provided the model parameters N, M, a_{ij}, $e_j(k)$, and π_i (*i.e.* finding $P(O \mid model)$).

If there is a one to one correspondence between the states and the observations, this problem can be solved with simple probability theory. The simple coin toss experiment described in section 3.1 illustrates the solution.

Solving problem one becomes more complex if the state sequence is not known. A straightforward approach to the solution involves listing every possible state sequence. A state sequence Q is defined as

$$Q = q_1, q_2, \ldots, q_T$$

In the above definition, q_1 is the initial state, q_2 is the state at the second time instance, *etc.* The probability of an observation sequence O given the state sequence Q and the model is given by:

$$\mathrm{P}(O \mid Q, model) = \prod_{t=1}^{T} \mathrm{P}(O_t \mid q_t, model)$$

In the above equation, T is the length of the sequences. The probability that the state sequence Q will occur with the given model parameters is:

$$\mathrm{P}(Q \mid model) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

The joint probability that the state sequence Q and the observation sequence O occur simultaneously is just the product of the above two equations:

$$\mathrm{P}(O, Q \mid model) = \mathrm{P}(O \mid Q, model)\, \mathrm{P}(Q \mid model)$$

Finally, to determine the probability that O was generated by the model, sum all of the joint probabilities for all of the possible state sequences:

$$\mathrm{P}(O \mid model) = \sum_{\forall Q} \mathrm{P}(O, Q \mid model)$$

The problem with the above approach is there are N^T possible state sequences that might have generated the observation sequence in question. As the sequences increase in length, the number of state paths increases exponentially and enumerating all possible paths becomes impractical. The forward algorithm is a more efficient solution to problem one, and is the subject of the next section.

In the weld defect classification problem presented in this thesis, the forward algorithm is needed to do the final classification of the flaw signals. The state transition probability, emission probability, and initial state probability matrices must first be estimated using the Baum-Welch algorithm (section 3.2.1.7). One set of model parameters must be estimated for each of the four types of defects than can occur in welds. For a given test sequence, the forward algorithm is used to calculate four scores, one for each defect model. These scores quantify the probability that the test sequence was generated by the model (*e.g.* the score $P(O \mid crk\ model)$ is the probability that the test sequence is a crack). The maximum of these four scores indicates the class of the test signal. The assignment goes to the defect class with the highest probability score.

3.2.1.2. The Forward Algorithm

The forward variable $f_t(i)$ is the probability of observing the partial sequence $O_1, O_2, \ldots O_t$ and state S_i at time t (*i.e.* $f_t(i) = P(O_1, O_2, \ldots, O_t, q_t = S_i \mid model)$). $P(O \mid model)$ is the probability that the complete sequence was generated by the model. To calculate $f_t(i)$ and

P(O | *model*), use the forward algorithm, which follows. Recall that in order to use the forward algorithm, the model parameters N, M, **A**, **E**, and π must be known.

1. Initialize the forward variable for each state i:

   $f_1(i) = \pi_i e_i(O_1)$ where $1 \le i \le N$

2. Calculate the forward variable vectors for each state j:

   $$f_{t+1}(j) = \left(\sum_{i=1}^{N} f_t(i) a_{ij} \right) e_j(O_{t+1}) \text{ where } 1 \le t \le T-1 \text{ and } 1 \le j \le N$$

3. Terminate the algorithm by calculating the probability that the complete sequence was generated by the model:

   $$\mathrm{P}(O \mid model) = \sum_{i=1}^{N} f_T(i)$$

The probabilities calculated by the forward algorithm can be very small, so it is advisable to do the calculations in log space to avoid underflow errors.

3.2.1.3. The Second Problem for Hidden Markov Models

The second problem for hidden Markov models is to determine the most likely state sequence when given an observation sequence and the model parameters N, M, **A**, **E**, and π.

An example of this problem is that of the occasionally dishonest casino [26]. The casino has a fair die and a loaded die that can be given to its patrons without their knowledge that a change has occurred. This situation can be described using a hidden Markov model where

state one means the gambler has been given a fair die and state two means the gambler is playing with a loaded die. The gambler does not know if the die he has been given is fair or loaded.

For any fair die (state one) the probabilities for all of the observations are equal, so the emission probabilities $e_1(1)$, $e_1(2)$, $e_1(3)$, $e_1(4)$, $e_1(5)$, and $e_1(6)$ are all $\frac{1}{6}$. For the loaded die (state two), there is a greater chance of observing a six, and the emission probabilities are: $e_2(1) = e_2(2) = e_2(3) = e_2(4) = e_2(5) = 0.1$ and $e_2(6) = 0.5$.

The state transition probability matrix is defined as the following:

$$\mathbf{A} = \begin{bmatrix} 0.95 & 0.05 \\ 0.1 & 0.9 \end{bmatrix}$$

This means that if the gambler currently is using a fair die, there is a 95% chance that the casino will give him the fair die for the next roll, and there is a 5% chance the casino will instead slip him a loaded die for the next roll. If the gambler currently has a loaded die, there is a 10% chance the casino will give him the loaded die again for the next roll, and a 90% chance they will switch to the fair die in the next roll.

Given this information and an observed sequence of several rolls, one can estimate the state sequence that is most likely to produce the observation sequence. This is accomplished using the Viterbi algorithm (Section 3.2.1.4). An example where this was done is shown below.

Table 3-1. Predicting state sequence using the Viterbi Algorithm

Observation	3	1	6	6	6
Prediction	Fair	Fair	fair	Loaded	Loaded

When the observations appear to be random selections of the six possible face values of the die, it is more likely that the die is the fair one. When several consecutive sixes are observed, it becomes more likely that the observations have been generated by the loaded die.

In the case of the weld defect classification problem, the states do not have special meanings. Each state has its own unique set of properties, but the states do not have a physical interpretation in this problem.

3.2.1.4. The Viterbi Algorithm

The Viterbi algorithm is very similar to the forward algorithm. In fact, the Viterbi algorithm is equivalent to performing the forward algorithm with the sum from step 3 replaced by maximization. The Viterbi variable $v_t(i)$ is the probability that the process was in state i when observation t was observed. P_t is the predicted state for observation t. Recall that to use the Viterbi algorithm, one must have an observation sequence and the model parameters N, M, $\mathbf{A}$, $\mathbf{E}$, and π.

1. Initialize the Viterbi variable for each state i:

   $v_t(i) = \pi_i e_i(O_1)$ where $1 \le i \le N$

2. Calculate the Viterbi variable vectors for each state j:

$$v_{t+1}(j) = \left(\sum_{i=1}^{N} v_t(i) a_{ij} \right) e_j(O_{t+1}) \text{ where } 1 \le t \le T-1 \text{ and } 1 \le j \le N$$

3. Determine P_t for each observation in the sequence by picking the state with the largest value of the Viterbi variable for that observation:

$P_t = \max(v_t(i))$ where $1 \le t \le T$ and $1 \le i \le N$

As with the forward algorithm, it is advisable to work in log space when using the Viterbi algorithm because the probabilities can be rather small and underflow errors can occur.

3.2.1.5. The Third Problem for Hidden Markov Models

The third problem for hidden Markov models is the most difficult. Only a set of observation sequences is provided, and the problem involves determining the model parameters **A**, **E**, and π. Recall from sections 3.1.1.1 and 3.1.1.3 that all the model parameters must be known in order to use the forward and Viterbi algorithms, so this may be the most important problem.

Since HMMs are a probabilistic method and the states are often not known, there is no method available for calculating the model parameters directly. These parameters must be estimated using a series of training sequences and an iterative process called the Baum-Welch algorithm (Section 3.2.1.7). To perform the Baum-Welch algorithm, the backward algorithm is required.

3.2.1.6. The Backward Algorithm

The backward algorithm is used to calculate the posterior state probabilities. It is similar to the forward algorithm, but begins at the end of the sequence. The backward variable $b_t(i)$ is defined as:

$$b_t(i) = \mathrm{P}(O_{t+1}, O_{t+2}, \ldots, O_T \mid q_t = S_i, model)$$

In other words, $b_t(i)$ is the probability of a partial observation sequence from a point $t + 1$ in the sequence to the end of the sequence given the process is in state S_i at time t and the model parameters. T is the length of the observation sequences. N is the number of states.

1. Initialize the backward variable for each state i:

$$b_T(i) = 1 \ \forall \ i \text{ where } 1 \le i \le N.$$

2. Calculate the backward variable vectors:

$$b_t(i) = \sum_{j=1}^{N} a_{ij} e_j(O_{t+1}) b_{t+1}(j) \text{ where } t = T-1, T-2, \ldots, 1 \text{ and } 1 \le i \le N.$$

As with the forward and Viterbi algorithms, it is advisable to perform the backward algorithm in log space since the probabilities can be very small.

3.2.1.7. The Baum-Welch Algorithm

The Baum-Welch algorithm, also known as the forward-backward algorithm, is an efficient method for estimating the parameters of a hidden Markov model. [26, 27]

1. Choose arbitrary initial values for the parameters **A**, **E**, and π.
2. Initialize variables $A_t(k, l)$ and $E_k(b)$ to zero. $A_t(k, l)$ is the expected number of times a_{kl} will be used. $E_k(b)$ is the expected number of times observation b will occur when the state is k.
3. For each sequence $j = 1, 2, \ldots, n$:
    a. Calculate the forward variable, $f_t(i)$, for sequence j using the forward algorithm, which was discussed in section 3.2.1.2.
    b. Calculate the backward variable, $b_t(i)$, for sequence j using the backward algorithm, which was discussed in section 3.2.1.6.
    c. Add the contribution of sequence j to $A_t(k, l)$ using the following equation:

$$A_t(k,l) = \frac{f_t(k)a_{kl}e_l(O_{t+1})b_{t+1}(l)}{\mathrm{P}(O \mid model)}$$

In the above equation, $A_t(k, l)$ is calculated for every combination of t, k, and l where t indicates the position in the observation sequence. $1 \le t \le T-1$ where T is the length of observation sequence j, k and l represent states ($1 \le k, l \le N$), and $\mathrm{P}(O \mid model)$ is the probability of the observation sequence as calculated by either the forward or backward algorithm.

    d. Add the contribution of sequence j to $E_k(b)$ using the following equation:

$$E_k(b) = \frac{f_t(k)b_t(k)}{\mathrm{P}(O \mid model)}$$

In the above equation $1 \le t \le T$ and $1 \le k \le N$.

4. Recompute the model parameters. The initial state probabilities are adjusted using the following equation:

   $\pi_k = E_k(1)\ \forall\ k$ where $1 \le k \le N$.

   The state transition probabilities are adjusted using the following equation:

$$a_{kl} = \frac{\sum_{t=1}^{T-1} A_t(k,l)}{\sum_{t=1}^{T} E_k(b)}\ \forall\ k, l \text{ where } 1 \le k, l \le N.$$

   The emission probabilities are adjusted using the following equation:

$$e_k(b) = \frac{\sum_{t=1 \ s.t. O_t = b}^{T} E_k(b)}{\sum_{t=1}^{T} E_k(b)}$$

5. Calculate the new log likelihood of the model.
6. Repeat steps two through five. Stop repeating this procedure when the change in log likelihood falls below a predefined threshold or when the maximum number of iterations is exceeded.

3.3. Classifying Weld Defects Using Hidden Markov Models

Four main data sets were available for this project: 2 MHz 60° signals, 2 MHz 70° signals, 5 MHz 60° signals, and 5 MHz 70° signals. Because these four data sets have dissimilar

properties, separate models must be developed for each set. For example, the model developed for 2 MHz 60° signals cannot be used to classify signals from any of the other three data sets. Additionally, using one of these data sets for the training data and another for validation would perform unpredictably.

For this project, 2 MHz 60° signals were used exclusively because these signals comprise the largest group of signals. This data set contains 4,881 cracks, 8,821 lacks of fusion, 2,635 porosities, and 2,519 slags. Half of the defects of each category were randomly selected for use as training data, and the other half were used as validation data.

Before model construction and classification can be performed, some signal processing is required. Each A-scan is gated to a length of 512. Gating allows one to concentrate only on the region of interest around the defect. Additionally, having all the signals be the same length makes computation easier. The gated A-scans are windowed using a Hamming window. Windowing with a hamming window deemphasizes the "tail" portions of the signals and emphasizes the "middle" of the signal where the defect is located. The signals are then aligned using the peak amplitude.

When one works with hidden Markov models, generally the observations are discrete. Quantization divides a large number of observations into a manageable set. For some applications, a continuous Markov model is more appropriate. [27] However, continuous Markov models are not addressed as a part of this thesis. Quantization reduces the number of observations that are possible. This simplifies the model and reduces computation time.

Two quantization schemes are considered in this thesis. The first scheme is based on the amplitudes of the A-scans. If the signal amplitude at a given time is within the range [-130, -120), then the observed symbol is 1. If the amplitude is within the range [-120, -110), then the observed symbol is 2. This quantization scheme continues in a similar manner, accommodating amplitudes up to 130 for a total of 26 possible observations. A 27^{th} possible observation is also included to accommodate any signals that do not fall into the range [-130, 130].

In many cases, a quantization scheme that captures the structure of the signals performs better in the classification phase. For this reason, a second quantization scheme based on slope was explored. Each A-scan was divided into sections, each of which was four time units long. The slope of each line segment was calculated and quantized. If the slope was within the range $\left[\frac{-\pi}{2}, \frac{-3\pi}{8}\right)$, then the observed signal was 1. If the slope was within the range $\left[\frac{-3\pi}{8}, \frac{-\pi}{4}\right)$, then the observation was 2. This quantization scheme continues in a similar manner to accommodate slopes up to $\frac{\pi}{2}$ for a total of eight possible observation symbols. A ninth symbol was also included to include any other slopes, but this symbol would indicate an error has occurred.

As a final preprocessing step, the A-scans are randomized to eliminate any possible relationship that may exist between signals based on order.

For this project, the publicly available Matlab program HTK, which is short for Hidden Markov Model Toolkit, was used. [28] Four models must be constructed, one for each type of defect. The only available information is the observation sequences, the ultrasonic scans, and the classification of each defect. Building the models involves using the Baum-Welch algorithm to estimate the model parameters **A**, **E**, and π. Using HTK, this can be accomplished by doing the following:

1. Initialize **A**, **E**, and π with random quantities using the following commands:

>> *prior1* = normalise(rand(*Q*,1));

>> *transmat1* = mk_stochastic(rand(*Q*,*Q*));

>> *obsmat1* = mk_stochastic(rand(*Q*,*O*));

Q is the number of states and *O* is the number of observation symbols. *Prior1* is the initial estimate of the initial state probability matrix π, *transmat1* is the initial estimate of the state transition probability matrix **A**, and *obsmat1* is the initial estimate of the emission probability matrix **E**.

2. Call the script learn_dhmm.m, which adjusts the initial model parameter estimates using the Baum-Welch algorithm. This is done with the following command line:

>> [*LL*, *prior2*, *transmat2*, *obsmat2*] = learn_dhmm(*data*, *prior1*, *transmat1*, *obsmat1*, *max_iter*);

This function call will need to be performed four times, once for each type of defect. Five quantities are passed to learn_dhmm(). *Data* is a set of training sequences, and is comprised of either all cracks, lacks of fusion, porosities, or slags. *Prior1*, *transmat1*, and *obsmat1* are the initial estimates of the model parameters introduced in step 1. *Max_iter* is the maximum

number of Baum-Welch iterations. Baum-Welch training will continue until the algorithm converges, meaning there ceases to be an improvement in the log likelihood, or until the maximum number of iterations has been reached. The algorithm has converged when the change in log-likelihood drops below a predefined threshold, 1×10^{-4} in this case. *Max_iter* was set high enough to allow the Baum-Welch algorithm to converge.

Learn_dhmm returns four quantities. *LL(t)* is the log-likelihood after iteration *t*. *Prior2*, *transmat2*, and *obsmat2* are the adjusted model parameters returned by the function.

To determine the optimal number of states, Q was varied.

Once models are constructed for each of the four defect types, classification can commence. For each test sequence, calculate a score for each defect model. Each score is the probability that the test sequence was generated by the model. The model that receives the highest score is the class the test sequences belongs to. Using HTK, the function log_lik_dhmm is called four times to calculate scores for the four defect types:

>> *crkprob* = log_lik_dhmm(*obs*, *crkprior*, *crktransmat*, *crkobsmat*);

>> *lofprob* = log_lik_dhmm(*obs*, *lofprior*, *loftransmat*, *lofobsmat*);

>> *porprob* = log_lik_dhmm(*obs*, *porprior*, *portransmat*, *porobsmat*);

>> *slgprob* = log_lik_dhmm(*obs*, *slgprior*, *slgtransmat*, *slgobsmat*);

The function log_lik_dhmm() takes four arguments. *Obs* is the observation sequence to be classified. *Crkprior* is the initial probability matrix for the crack model, *crktransmat* is the state transition probability matrix for the crack model, and *crkobsmat* is the emission

probability matrix for the crack model. Likewise, these parameters are also defined for lacks of fusion, porosities, and slags. The function returns one quantity which is the score for that model. (*E.g.* if *crkprob* is highest score, assign the observation sequence the class crack.) This classification procedure is performed for every sequence in both the training and validation data sets.

3.3. Support Vector Machines

Classification of the weld defects with support vector machines (SVMs) was performed using the publicly available program SVMLight. [29] Because SVMs do not require discrete inputs, the unquantized data was used for training and classification.

SVMLight consists of two main programs. The first is svm_learn.exe, which is used first with a set of training defect samples for the purpose of generating the model. The second program is svm_classify which uses the model generated by svm_learn to classify a set of defect samples.

Training is accomplished using the following command:

> svm_learn *training_data.dat model.dat*

The first argument, *training_data.dat*, is a file containing several defect samples for training the model. This input file must be formatted such that each row contains a class label for a defect signal, either -1 or +1, followed by the signal. The second argument, *model.dat*, is a file for storing the model that is generated by svm_learn.

Classification is accomplished using:

> svm_classify *testing_data.dat model.dat classification.dat*

The file *testing_data.dat* contains a set of defect signals that are to be classified. The file *model.dat* is the file generated using svm_learn. Finally, *classification.dat* is a file in which the classification decisions are stored. The sign of the number stored in *classification.dat* determines the class of the corresponding defect signal.

The training and validation data used for SVM classification is identical to that used for HMM classification. To avoid overtraining for the planar class, SVM classification was also performed using 1000 training and 1000 validation samples from each defect class.

Chapter 4. Results

4.1. HMM and SVM Results Overview

The classification results are presented in a series of tables called confusion matrices. Each row in the table represents the class distribution of one defect type. The first half of each cell is the percent for the training data, and the second half is for the validation data. For example, Table 4.1 indicates that 0% of the crack training sequences and 0% of the crack validation sequences were correctly classified as cracks, 11% of the crack training sequences and 9% of the validation training sequences were classified as lacks of fusion, 1% of the crack training and validation sequences were classified as porosities, and 88% of the crack training sequences and 90% of the crack validation sequences were classified as slags. Similar tables are provided for the two class problem. Overall classification accuracy is provided in the bottom cell of each confusion matrix.

4.2. HMM Classification Results for Two States

Table 4-1. HMM classification results for 2 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	0%	0%	11%	9%	1%	1%	88%	90%
LOF (4410)	1%	1%	45%	44%	0%	0%	54%	55%
POR (1317)	0%	0%	1%	1%	1%	1%	98%	98%
SLG (1260)	0%	0%	22%	24%	0%	0%	78%	76%
Overall accuracy: 31% training, 30% validation								

	Planar		Volumetric	
PLN (6850)	34%	32%	66%	68%
VOL (2577)	11%	12%	89%	88%
Overall accuracy: 59% training, 58% validation				

Table 4-2. HMM classification results for 2 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	65%	65%	0%	0%	35%	35%	0%	0%
LOF (4410)	83%	83%	0%	0%	17%	17%	0%	0%
POR (1317)	35%	38%	0%	0%	65%	62%	0%	0%
SLG (1260)	84%	82%	0%	0%	16%	18%	0%	0%
Overall accuracy: 33% training, 32% validation								

	Planar		Volumetric	
PLN (6850)	77%	77%	23%	23%
VOL (2577)	59%	60%	41%	40%
Overall accuracy: 57% training, 57% validation				

4.3. HMM Classification Results for Three States

Table 4-3. HMM classification results for 3 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	85%	85%	15%	15%	0%	0%	0%	0%
LOF (4410)	40%	43%	52%	49%	0%	0%	8%	8%
POR (1317)	90%	88%	10%	12%	0%	0%	0%	0%
SLG (1260)	63%	62%	37%	38%	0%	0%	0%	0%
Overall accuracy: 34% training, 34% validation								

	Planar		Volumetric	
PLN (6850)	95%	95%	5%	5%
VOL (2577)	100%	100%	0%	0%
Overall accuracy: 48% training, 48% validation				

Table 4-4. HMM classification results for 3 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	98%	96%	0%	0%	1%	2%	1%	2%
LOF (4410)	96%	97%	0%	0%	0%	0%	4%	3%
POR (1317)	100%	100%	0%	0%	0%	0%	0%	0%
SLG (1260)	96%	96%	0%	0%	0%	0%	4%	4%
Overall accuracy: 25% training, 25% validation								

	Planar		Volumetric	
PLN (6850)	97%	97%	3%	3%
VOL (2577)	98%	98%	2%	2%
Overall accuracy: 49% training, 50% validation				

4.4. HMM Classification Results for Four States

Table 4-5. HMM classification results for 4 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	4%	4%	0%	0%	10%	10%	86%	86%
LOF (4410)	0%	0%	5%	5%	1%	1%	94%	94%
POR (1317)	0%	0%	0%	0%	23%	21%	77%	79%
SLG (1260)	1%	1%	1%	0%	0%	2%	98%	97%
Overall accuracy: 32% training, 32% validation								

	Planar		Volumetric	
PLN (6850)	5%	5%	95%	95%
VOL (2577)	1%	1%	99%	99%
Overall accuracy: 52% training, 52% validation				

Table 4-6. HMM classification results for 4 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	36%	33%	20%	21%	31%	33%	13%	13%
LOF (4410)	12%	13%	61%	59%	9%	11%	18%	17%
POR (1317)	23%	22%	10%	10%	60%	61%	7%	7%
SLG (1260)	23%	20%	40%	39%	11%	15%	26%	26%
Overall accuracy: 45% training, 45% validation								

	Planar		Volumetric	
PLN (6850)	67%	66%	33%	34%
VOL (2577)	48%	45%	52%	55%
Overall accuracy: 58% training, 59% validation				

4.5. HMM Classification Results for Five States

Table 4-7. HMM classification results for 5 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	14%	12%	2%	2%	84%	86%	0%	0%
LOF (4410)	14%	14%	31%	30%	55%	56%	0%	0%
POR (1317)	2%	3%	0%	0%	98%	97%	0%	0%
SLG (1260)	25%	25%	11%	11%	64%	64%	0%	0%
Overall accuracy: 35% training, 35% validation								

	Planar		Volumetric	
PLN (6850)	34%	34%	66%	66%
VOL (2577)	19%	19%	81%	81%
Overall accuracy: 56% training, 55% validation				

Table 4-8. HMM classification results for 5 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	28%	29%	4%	4%	19%	19%	49%	48%
LOF (4410)	9%	10%	9%	8%	6%	7%	76%	75%
POR (1317)	14%	16%	2%	2%	39%	38%	45%	44%
SLG (1260)	13%	13%	5%	6%	6%	6%	76%	76%
Overall accuracy: 38% training, 38% validation								

	Planar		Volumetric	
PLN (6850)	23%	23%	77%	77%
VOL (2577)	17%	18%	83%	82%
Overall accuracy: 54% training, 54% validation				

4.6. HMM Classification Results for Six States

Table 4-9. HMM classification results for 6 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	55%	55%	26%	27%	17%	16%	2%	2%
LOF (4410)	31%	31%	55%	54%	9%	10%	5%	5%
POR (1317)	40%	37%	16%	19%	44%	44%	0%	0%
SLG (1260)	45%	46%	33%	32%	13%	13%	9%	9%
Overall accuracy: 41% training, 40% validation								

	Planar		Volumetric	
PLN (6850)	84%	84%	16%	16%
VOL (2577)	67%	67%	33%	33%
Overall accuracy: 58% training, 58% validation				

Table 4-10. HMM classification results for 6 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	78%	79%	5%	5%	11%	10%	6%	6%
LOF (4410)	54%	56%	26%	25%	10%	10%	10%	9%
POR (1317)	78%	80%	1%	1%	17%	14%	4%	5%
SLG (1260)	56%	54%	9%	9%	14%	14%	21%	20%
Overall accuracy: 35% training, 34% validation								

	Planar		Volumetric	
PLN (6850)	81%	82%	19%	18%
VOL (2577)	72%	74%	28%	26%
Overall accuracy: 55% training, 54% validation				

4.7. HMM Classification Results for Seven States

Table 4-11. HMM classification results for 7 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	28%	28%	6%	6%	48%	47%	18%	19%
LOF (4410)	20%	21%	41%	40%	21%	20%	18%	19%
POR (1317)	6%	6%	0%	2%	85%	82%	9%	10$
SLG (1260)	31%	33%	20%	19%	32%	30%	17%	18%
Overall accuracy: 43% training, 42% validation								

	Planar		Volumetric	
PLN (6850)	52%	51%	48%	49%
VOL (2577)	28%	29%	72%	71%
Overall accuracy: 59% training, 59% validation				

Table 4-12. HMM classification results for 7 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	7%	7%	3%	3%	58%	58%	32%	32%
LOF (4410)	7%	7%	18%	18%	23%	24%	52%	51%
POR (1317)	3%	3%	1%	0%	80%	84%	16%	13%
SLG (1260)	8%	9%	4%	6%	48%	46%	40%	39%
Overall accuracy: 37% training, 37% validation								

	Planar		Volumetric	
PLN (6850)	20%	20%	80%	80%
VOL (2577)	8%	9%	92%	91%
Overall accuracy: 55% training, 55% validation				

4.8. HMM Classification Results for Eight States

Table 4-13. HMM classification results for 8 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	28%	29%	2%	2%	48%	47%	22%	22%
LOF (4410)	13%	14%	33%	32%	17%	17%	37%	37%
POR (1317)	8%	8%	0%	0%	88%	86%	4%	6%
SLG (1260)	17%	18%	10%	10%	34%	31%	39%	41%
Overall accuracy: 47% training, 47% validation								

	Planar		Volumetric	
PLN (6850)	41%	41%	59%	59%
VOL (2577)	17%	18%	83%	82%
Overall accuracy: 61% training, 60% validation				

Table 4-14. HMM classification results for 8 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	53%	53%	23%	23%	6%	7%	18%	17%
LOF (4410)	22%	24%	52%	50%	5%	5%	21%	21%
POR (1317)	48%	51%	9%	7%	16%	17%	27%	25%
SLG (1260)	22%	20%	26%	27%	12%	11%	40%	42%
Overall accuracy: 40% training, 40% validation								

	Planar		Volumetric	
PLN (6850)	74%	75%	26%	25%
VOL (2577)	53%	53%	47%	47%
Overall accuracy: 61% training, 61% validation				

4.9. HMM Classification Results for Nine States

Table 4-15. HMM classification results for 9 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	62%	61%	10%	10%	22%	23%	6%	6%
LOF (4410)	36%	37%	39%	48%	6%	6%	9%	9%
POR (1317)	44%	44%	1%	1%	54%	54%	1%	1%
SLG (1260)	51%	47%	23%	25%	13%	14%	13%	14%
Overall accuracy: 44% training, 44% validation								

	Planar		Volumetric	
PLN (6850)	80%	80%	20%	20%
VOL (2577)	59%	59%	41%	41%
Overall accuracy: 60% training, 60% validation				

Table 4-16. HMM classification results for 9 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	33%	34%	1%	1%	52%	52%	14%	13%
LOF (4410)	35%	35%	13%	12%	25%	26%	27%	27%
POR (1317)	23%	23%	0%	1%	68%	68%	9%	8%
SLG (1260)	25%	28%	2%	2%	35%	34%	38%	37%
Overall accuracy: 38% training, 38% validation								

	Planar		Volumetric	
PLN (6850)	43%	43%	57%	57%
VOL (2577)	25%	26%	75%	74%
Overall accuracy: 58% training, 58% validation				

4.10. HMM Classification Results for Ten States

Table 4-17. HMM classification results for 10 states, amplitude based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	19%	20%	1%	1%	53%	52%	27%	27%
LOF (4410)	7%	8%	8%	7%	17%	16%	68%	69%
POR (1317)	4%	5%	0%	0%	90%	87%	6%	8%
SLG (1260)	11%	12%	1%	1%	32%	31%	56%	56%
Overall accuracy: 43% training, 42% validation								

	Planar		Volumetric	
PLN (6850)	17%	17%	83%	83%
VOL (2577)	8%	9%	92%	91%
Overall accuracy: 55% training, 55% validation				

Table 4-18. HMM classification results for 10 states, slope based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	21%	21%	1%	1%	39%	41%	39%	37%
LOF (4410)	22%	22%	7%	6%	27%	27%	44%	45%
POR (1317)	9%	9%	0%	0%	54%	56%	37%	35%
SLG (1260)	10%	11%	1%	1%	21%	23%	68%	65%
Overall accuracy: 37% training, 37% validation								

	Planar		Volumetric	
PLN (6850)	26%	26%	74%	74%
VOL (2577)	10%	10%	90%	90%
Overall accuracy: 58% training, 57% validation				

4.11. Classification Using Combined Models

To attain better classification results, models combining the best results were considered. For each defect class, the model that produced the best results was identified. For the amplitude-based quantization scheme, best classification for cracks was achieved with the three state model, best lack of fusion results with the six state model, best porosity results with the five state model, and best slag results with the four state model. Previously, the probability scores for a given test sequence were all determined using models for each of the four defect classes and the same number of states for each model. Using combined models, the probability that the test sequence is a crack is determined using the model that performed best for classifying cracks. Similarly, the best lack of fusion, porosity, and slag models are used to calculate the probabilities that test sequences belong to those classes. This combined model is scheme 1, and the results for scheme 1 are shown in Table 4.19.

A similar approach was taken with the data that underwent the slope-based quantization scheme. Best crack classification results were achieved with three states, lack of fusion with four states, porosity with seven states, and slag with five states. This combined model is scheme 2, and the results for scheme 2 are shown in Table 4.21.

The best models for the two class problem were also considered. The models that performed best at distinguishing planar and volumetric flaws were identified. For the amplitude-based quantization scheme, the three state model performed best at classifying planar flaws and the four state model performed best at classifying volumetric flaws. This

combined model is scheme 3, and its results are shown in Table 4.20. For the slope-based quantization scheme, the three state model was best for planar flaws and the seven state model was best for the volumetric flaws. This combined model is scheme 4, and its results are shown in Table 4.22.

Table 4-19. HMM classification results for scheme 1, amplitude-based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	2%	3%	33%	32%	65%	65%	0%	0%
LOF (4410)	1%	2%	68%	67%	29%	29%	2%	2%
POR (1317)	1%	1%	26%	30%	73%	69%	0%	0%
SLG (1260)	3%	4%	51%	50%	44%	44%	2%	2%
Overall accuracy: 43% training, 42% validation								

	Planar		Volumetric	
PLN (6850)	57%	57%	43%	43%
VOL (2577)	40%	42%	60%	58%
Overall accuracy: 58% training, 57% validation				

Table 4-20. HMM classification results for scheme 3, amplitude-based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	59%	58%	8%	7%	8%	8%	25%	27%
LOF (4410)	24%	25%	45%	44%	1%	1%	30%	30%
POR (1317)	54%	55%	1%	1%	16%	13%	29%	31%
SLG (1260)	40%	42%	23%	24%	0%	1%	37%	33%
Overall accuracy: 43% training, 42% validation								

	Planar		Volumetric	
PLN (6850)	68%	68%	32%	32%
VOL (2577)	59%	61%	41%	39%
Overall accuracy: 60% training, 60% validation				

Table 4-21. HMM classification results for scheme 2, slope-based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	3%	3%	3%	3%	84%	83%	10%	11%
LOF (4410)	1%	0%	21%	21%	53%	54%	25%	25%
POR (1317)	0%	0%	1%	1%	95%	97%	4%	2%
SLG (1260)	0%	0%	4%	4%	79%	78%	17%	18%
Overall accuracy: 26% training, 26% validation								

	Planar		Volumetric	
PLN (6850)	16%	16%	84%	84%
VOL (2577)	2%	3%	98%	97%
Overall accuracy: 38% training, 38% validation				

Table 4-22. HMM classification results for scheme 4, slope-based

	Crack		Lack of Fusion		Porosity		Slag	
CRK (2440)	2%	3%	0%	0%	52%	54%	46%	43%
LOF (4410)	3%	2%	0%	0%	44%	44%	54%	54%
POR (1317)	0%	0%	0%	0%	61%	62%	39%	38%
SLG (1260)	1%	0%	0%	0%	27%	29%	72%	71%
Overall accuracy: 19% training, 19% validation								

	Planar		Volumetric	
PLN (6850)	2%	2%	98%	98%
VOL (2577)	1%	0%	99%	100%
Overall accuracy: 28% training, 29% validation				

4.12. Results of Principal Component Analysis

The principal components (PCs) of the data were determined using Morph3D. [30] One goal of principal component analysis (PCA) is attaining a data set that is reduced in dimensionality before proceeding with classification. For this thesis, it was used to gage the validity of using linear classifiers. For the non-quantized data, 41 PCs are required to account for 90% of the variability in the data. Figure 4.1 shows the cumulative principal components for the non-quantized data.

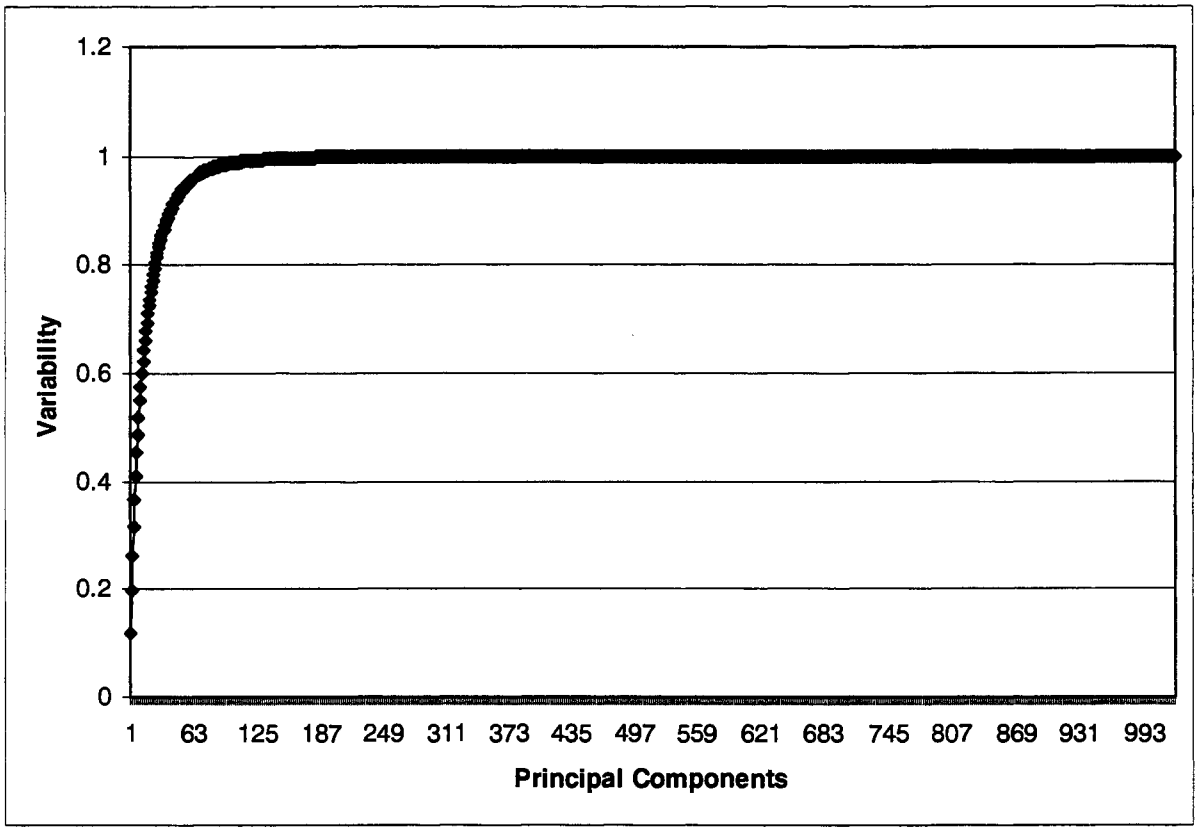

Figure 4-1. Principal Components for Non-quantized Data

PCA was also performed on the data quantized with the amplitude-based scheme. For this quantization scheme, 126 PCs are required to account for 90% of the variability in the data.

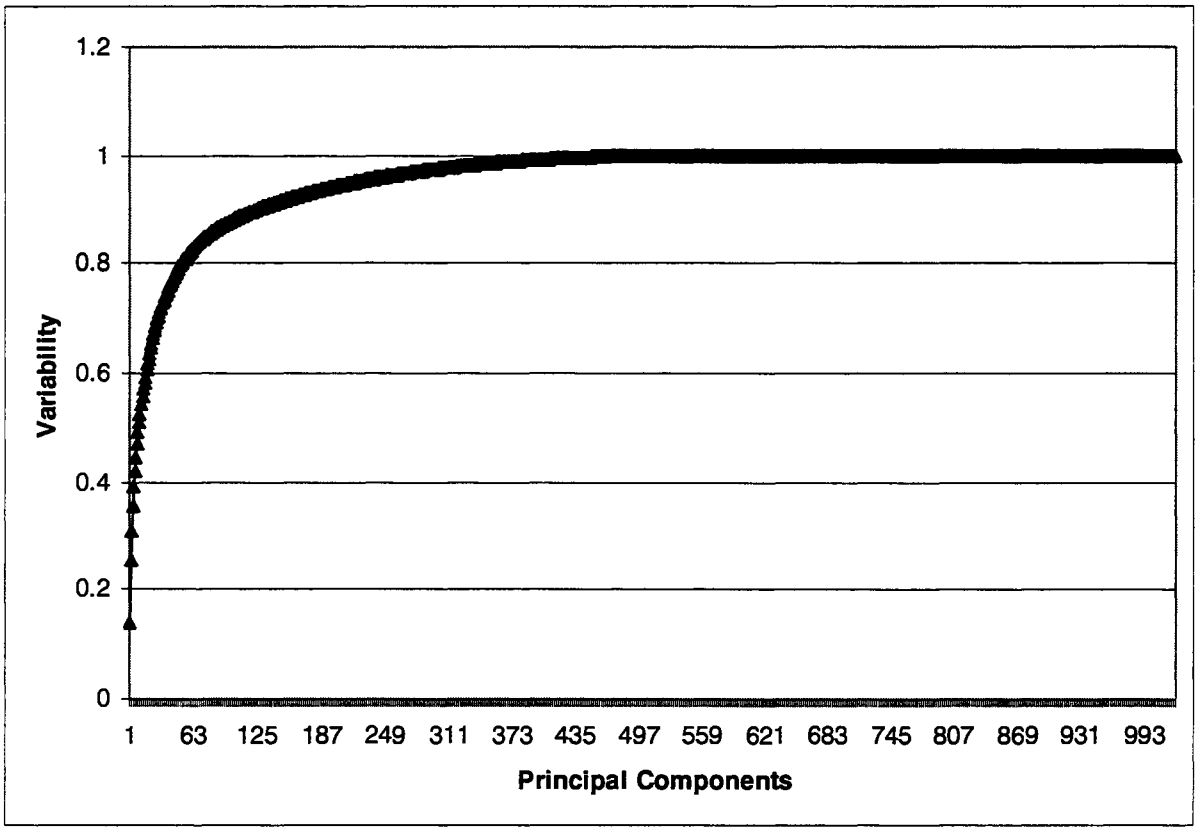

Figure 4-2. Principal Components for Quantized Data

4.13. Results of Support Vector Machine Classification

Support vector machine (SVM) classification was applied to the non-quantized data for the purpose of comparison to HMMs. The same training and validation data were used for SVMs as was used in HMMs. The results are shown in Table 4.23.

Table 4-23. Initial SVM classification results

	Planar		Volumetric	
PLN (6850)	95%	95%	5%	5%
VOL (2577)	63%	63%	37%	37%
Overall accuracy: 79% training, 79% validation				

Because many more planar sequences were used for training and validation, there appears to have been some overtraining. To avoid the bias toward planar flaws that appears in the initial SVM classification results, a second approach to SVM classification was taken. In this second approach, 1000 samples of each defect type were used for both training and validation. The classification results for this approach are shown in Table 4-24.

Table 4-24. SVM classification results

	Planar		Volumetric	
PLN (2000)	75%	74%	25%	26%
VOL (2000)	27%	29%	73%	71%
Overall accuracy: 74% training, 73% validation				

Chapter 5. Discussion

In this thesis, the use of hidden Markov models (HMMs) and support vector machines (SVMs) as classifiers for weld defects was investigated. Principal component analysis (PCA) was also used to gage the validity of linear classifiers that have been used thus far almost exclusively by the Materials Assessment and Research Group (MARG) at Iowa State University. This chapter contains conclusions regarding the effectiveness of these approaches as well as recommendations for future work.

5.1. Comparison of HMM Approaches

Since the proper number of states necessary to summarize the data was not known *a priori*, models were generated in the range of two to ten states. Figure 5.1 shows the overall accuracies that were achieved for the four class problem. Figure 5.2 shows the same for the two class problem.

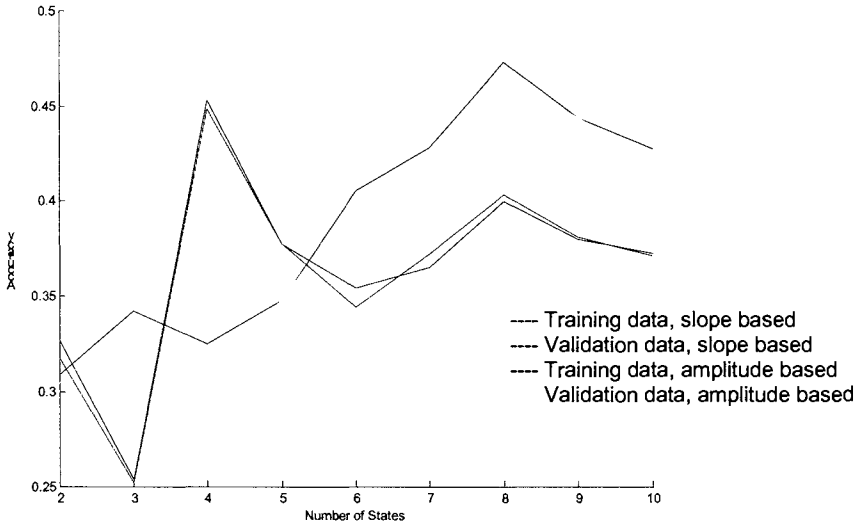

Figure 5-1. Classification Results for the Four Class Problem

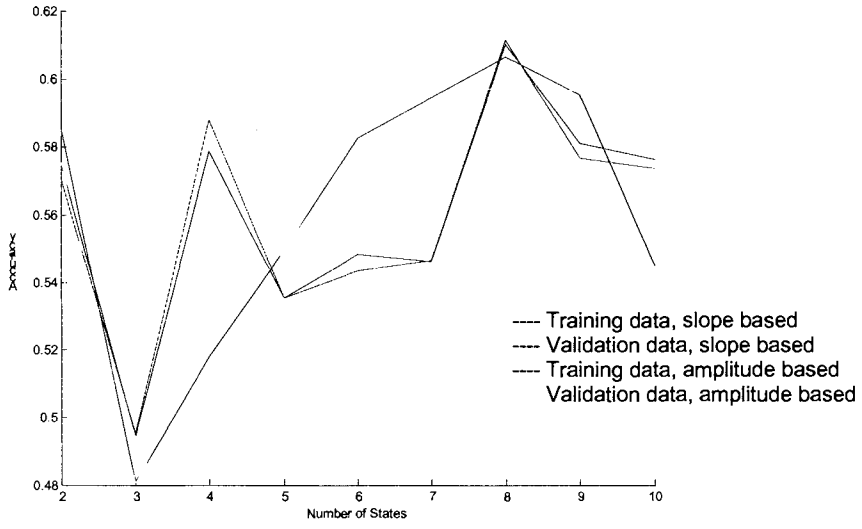

Figure 5-2. Classification Results for the Two Class Problem

The accuracy fluctuates; there does not appear to be a direct relationship between the number of states and the classification accuracy. At best, HMMs are 47% accurate for both training and validation data for the amplitude-based quantization scheme and the four class problem. Accuracy for the slope-based scheme and the four class problem is 40% for both training and validation data. When considering the two class problem, accuracy is approximately 60% for both quantization schemes. Peak performance for both four and two class problems occurs with eight states. There do not seem to be significant differences between the results of the amplitude and slope based quantization schemes. The amplitude-based quantization scheme actually performs slightly better, contradicting the hypothesis that the slope based scheme would perform better.

The combined schemes described in section 4.11 utilize the best performing individual models. Previously, classification was done using four models, one for each defect type, in which each model had the same number of hidden states. For the combined schemes, the models for cracks, lacks of fusion, porosities, and slags could all have different numbers of states. It was hypothesized that using the best performing model for each defect class would produce better classification results. This was not the case. For the amplitude-based quantization scheme, 40% accuracy was achieved for the four class problem, and 60% accuracy was achieved for the two class problem. A decrease in accuracy occurred for the slope-based quantization scheme in which 26% accuracy was achieved for the four class problem and 38% accuracy was achieved for the two class problem.

5.2. Comparison to Principal Component Analysis

The goal of PCA is to reduce the dimensionality of the data while still retaining the important distinguishing features. The PCA results did not indicate that much of a reduction in the dimensionality of the weld defect data set is possible without non-linear transformations. For the non-quantized data, 41 principal components (PCs) are required to account for 90% of the variability in the data. For the data quantized with the amplitude-based scheme, 126 PCs are required to account for 90% of the variability in the data. While these results do not imply that the data is not separable, they do suggest that classification schemes that rely on linear separation will not work without performing transformations. This may explain why the relatively simple probabilistic models developed using HMMs did not work well. Also, the decision boundary between the classes will certainly be a complicated non-linear function. This explains why the multilayer perceptron algorithm has not produced acceptable results. Prior work in this area by a former MARG researcher indicated that twelve PCs provide nearly perfect classification of A-scans. [8] However, data used in that research was carefully hand selected, unlike the data used in this research.

5.3. Comparison of HMMs to Support Vector Machines

The publicly available program SVMLight [29] was used to classify the defect signals using support vector machines (SVMs). For the training and validation data sets, 79% correct classification was achieved. While these results are encouraging, closer examination of the confusion matrix indicated a bias toward the planar flaws. Accuracy for the planar flaws was 95%, but only 37% accuracy was achieved for the volumetric flaws. This was attributed to

the fact that 6850 planar flaws were used for both training and validation, but only 2577 volumetric flaws were used. This caused the SVM to overfit and classify most signals as planar. To remedy this situation, new training and validation data sets were sampled from the original data. One thousand of each of the four defect types were randomly chosen for training data. One thousand more of each type of defect were randomly chosen for validation data. This more balanced approach also demonstrated encouraging results, attaining 75% accuracy.

5.4. Recommendations for Future Work

At best, HMMs were able to classify 60% of the signals correctly. SVMs achieved 75% correct classification. These results are not spectacular, but many important lessons can be learned from analysis of the results.

While the accuracy of the HMM classification was low, it is possible that the results might improve with an increased number of states. The plots in Figures 5.1 and 5.2 do not imply an upward trend; however it is possible that the lower accuracy attained after eight states is only a local minima. As the number of states increases, the computation time required to develop the models from the training data increases. For this reason, a small selection of training samples should be used to develop higher state models. If these tests indicate that more states improve the classification, additional training time may prove to be worthwhile.

Alternatively, the use of SVMs may prove to be a more efficient and more fruitful approach. The use of HMMs involves estimation of a large number of unknown parameters. Additionally, HMMs are limited by the Markovian assumption, meaning that HMMs assume that the current state depends only on one predecessor state. This assumption is often invalid with complex data such as the weld defect data used in this research. SVMs are not so limited, and are capable of generating complex decision boundaries. Training and classification of the same weld defect data using SVMs required much less time, and the results were more accurate.

There is evidence that the data can be classified with an acceptable level of accuracy. However, these results may not be achievable using HMMs. Future researchers on this project may wish to focus their efforts on improving upon the SVM results, which yield more accurate results and are more computationally efficient than HMMs.

References

[1] C.G. Politt. Radiographic Sensitivity. *British Journal of NDT*, vol. 4, no. 3, pp. 71-80, September 1962.

[2] C.R.A. Schneider, I.J. Munns, G.A. Georgion, R.K. Chapman, G.S. Woodcock. Modelling the Radiography of Thick-Section Welds. *IEE Colloquium on New Applications for Non-Destructive Testing*, January 29, 1999.

[3] The Institution of Metallurgists. 1983. *Ultrasonic Non-Destructive Testing*. The Chameleon Press Limited.

[4] Department of Defense. 1974. *Military Standardization Handbook Ultrasonic Testing*. MIL-HDBK-726.

[5] H.A. Crostack, M. Maass. Non-Destructive Evaluation of Seams of Laser Welded Tools Using the CS-Eddy Current Technique. *Proceedings of the 24th Annual Conference of the IEEE*, vol. 4, pp. 2279–2283, 1998.

[6] J.W.H Tsai, M. Nathanson, R. Kimball, R. Geiss, T. Logan, T. Nguyen. Non-Destructive Defect Detection Scheme Using Kerr-Channel Optical Surface Analyzer. *IEEE Transactions on Magnetics*, vol. 37, no. 4, part 1, pp. 1957-1959, July 2001.

[7] D. O'Conchuir, D. McCurdy, V. Casey. Survey of Non-Destructive Inspection Methods for Solder Joint Integrity. *Proceedings of the IEEE*, vol. 3, pp. 1268-1275, 1991.

[8] R. Volesky. 2001. *Ultrasonic Signal Processing and Classification Using Principal Component Analysis*. Master's thesis, Iowa State University.

[9] D. Vray, M. Krueger, G. Gimenez. Synthetic Aperture-Based Beam Compression for Intravascular Ultrasound Imaging. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 48, no. 1, January 2001.

[10] L. Udpa. Personal correspondence. 2000-2002.

[11] R.C. Gonzalez, R.E. Woods. 2002. *Digital Image Processing.* Addison-Wesley Publishing Co.

[12] A.V. Oppenheim, A.S. Willsky, H. Nawab. 1997. *Signals and Systems.* Prentice Hall.

[13] S.J. Orfandis. 1996. *Introduction to Signal Processing*. Prentice Hall.

[14] J.G. Proakis and D.G. Manolakis. 1996. *Digitial Signal Processing Principles, Algorithms, and Applications.* Prentice Hall.

[15] B.B. Hubbard. 1998. *The World According to Wavelets.* A K Peters, Ltd.

[16] C.S. Burrus, R. Gopinath, H. Guo. 1998. *Introduction to Wavelets and Wavelet Transform.* Prentice-Hall.

[17] Y. Li, H.H. Szu, Y. Sheng, and H.J. Caulfield. Wavelet Processing and Optics. *Proceedings of the IEEE*, vol. 84, no. 5, pp. 720-732, May 1996.

[18] K.Y. Yeung and W.L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, vol. 17, no. 9, pp. 763-774, 2001.

[19] G.H. Dunteman. 1989. *Principal Components Analysis.* SAGE Publications Inc.

[20] I.T. Jolliffe. 1986. *Principal Component Analysis.* Springer-Verlag.

[21] M. Nadler and E.P. Smith. 1993. *Pattern Recognition Engineering.* John Wiley & Sons, Inc.

[22] P. Ramuhalli. 1998. *Automatic signal classification systems using fuzzy ARTMAP networks.* Master's thesis, Iowa State University.

[23] D.F. Specht. Probabilistic Neural Networks. *Neural Networks*, vol. 3, pp. 109-118, 1990.

[24] C.J.C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, December 1998.

[25] Y. Feng, S.Y. Lun, L. Di, L.Y. Zong. Application of Support Vector Machines to Quality Monitoring in Robotized Arc Welding. *Proceedings of the 2002 International Joint Conference on Neural Networks*, vol. 3, pp. 2321-2326, 2002.

[26] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. 1998. *Biological Sequence Analysis.* Cambridge University Press.

[27] L.R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257 – 286, Feb. 1989.

[28] HTK Hidden Markov Model Toolkit. http://htk.eng.cam.ac.uk/

[29] T. Joachims. SVMLight Support Vector Machine. http://svmlight.joachims.org/.

[30] K. Vander Velden. Morph3D: a program for protein morphology. Iowa State University, unpublished work.